

Sparse Partially Linear Additive Models

Yin Lou¹ Jacob Bien¹ Rich Caruana² Johannes Gehrke¹
¹Cornell University ²Microsoft Research
{yinlou, johannes}@cs.cornell.edu
jbien@cornell.edu rcaruana@microsoft.com

July 18, 2014

Abstract

The generalized partially linear additive model (GPLAM) is a flexible and interpretable approach to building predictive models. It combines features in an additive manner, allowing them to have either a linear or nonlinear effect on the response. However, the assignment of features to the linear and nonlinear groups is typically assumed known. Thus, to make a GPLAM a viable approach in situations in which little is known *a priori* about the features, one must overcome two primary model selection challenges: deciding which features to include in the model and determining which features to treat nonlinearly. We introduce sparse partially linear additive models (SPLAMs), which combine model fitting and *both* of these model selection challenges into a single convex optimization problem. SPLAM provides a bridge between the Lasso and sparse additive models. Through a statistical oracle inequality and thorough simulation, we demonstrate that SPLAM can outperform other methods across a broad spectrum of statistical regimes, including the high-dimensional ($p \gg N$) setting. We develop efficient algorithms that are applied to real data sets with half a million samples and over 45,000 features with excellent predictive performance.

1 Introduction

Generalized partially linear additive models (GPLAMs) [1] provide an attractive middle ground between the simplicity of generalized linear models (GLMs) [2] and the flexibility of generalized additive models (GAMs) [3]. Given a data set $\{(x_i, y_i)\}_{i=1}^N$, a GPLAM relates the conditional mean of the response, y_i , to the p -dimensional predictor vector, x_i , using a known link function, g :

$$g(E[y_i|x_i]) = \sum_{j \in \mathcal{N}} f_j(x_{ij}) + \sum_{j \in \mathcal{L}} w_j x_{ij}. \quad (1)$$

The features in \mathcal{N} contribute to the model in a nonlinear fashion while the features in \mathcal{L} contribute in a linear fashion. A GLM treats all features as being in \mathcal{L} and may therefore be biased when nonlinear effects are present; on the other extreme, a GAM treats all features as being in \mathcal{N} , which incurs unnecessary variance for the features that should be treated as linear. GPLAMs are a popular tool for data analysis in multiple fields including economics [4, 5] and biology [6, 7].

A major obstacle to using GPLAMs in machine learning problems such as text categorization is that in large-scale problems one rarely knows *a priori* which features should be assigned to \mathcal{N} and \mathcal{L} . A further challenge is in deciding which features should be excluded from the model entirely. The goal of this paper is to make GPLAMs a viable tool in the machine learner's toolkit. To do so, we must overcome two model-selection challenges: automatically deciding which features are at all relevant in the model and deciding which of those features should be fit linearly versus nonlinearly.

In the context of GAMs (where \mathcal{L} is taken to be empty), the sparse additive model (SpAM) is a useful framework for performing feature selection on \mathcal{N} [8]. From the perspective of a GPLAM, SpAM takes an

“all-in” or “all-out” approach to feature selection. In this work, we introduce the sparse partially linear additive model (SPLAM) that provides the finer-grained selection demanded by a GPLAM. SPLAMs build on the SpAM framework, providing a natural bridge between the ℓ_1 -penalized GLM and SpAM, thereby reaping many of the benefits enjoyed by both of these methods.

Failing to account for exactly linear features is disadvantageous on interpretability, statistical, and computational grounds. As a motivating example, consider a situation in which $p = 1,000$, $|\mathcal{N}| = 5$, and $|\mathcal{L}| = 295$. Assuming the correct set of features is selected, SpAM would include 300 features. From an interpretability standpoint, one would have to manually inspect the 300 nonparametric fits to reveal that only 5 features are effectively nonlinear (295 of them would appear nearly, but not exactly, linear such as in Figure 1 (d)); statistically, a price is paid in variance for the many nearly-linear features; and, computationally, such a model is wasteful both in terms of memory and speed for making future predictions.

In the last several years, a number of methods have been proposed to address various aspects of this problem. In [9], a bootstrap-based test is developed to determine the linearity of a component. In [10], the authors use a group MCP penalty to decide which features should be linear versus nonlinear, but features may not be completely excluded from the model. In [11], an algorithm is developed that iterates between two optimization problems, one that decides which nonlinear features should be made linear and the other that decides which linear features should be set to zero. An alternative approach to SpAM is the component selection and smoothing operator (COSSO) method, which uses unsquared reproducing kernel Hilbert space (RKHS) norm penalties [12]. The linear and nonlinear discoverer extends COSSO to the GPLAM setting [13]. Relatedly, [6] combine smoothness and sparsity SCAD-based penalties for a similar purpose. None of the above methods are geared for use in high-dimensional data in terms of statistical theory or computation. Several other methods are geared toward the high-dimensional setting but do not perform both model selection tasks. For example, in [14, 15, 16], methods are developed to perform feature selection for the set \mathcal{L} while assuming that the set \mathcal{N} is known; [17, 18] perform feature selection on both \mathcal{L} and \mathcal{N} individually but assume an initial partition of the features into those potentially in \mathcal{L} and those potentially in \mathcal{N} .

By contrast, SPLAM is designed for large-scale datasets (for example, we apply it to a dataset with $p = 47,236$). SPLAM is formulated through a single convex optimization problem that admits an efficient algorithm and strong theoretical properties even in the $p \gg N$ setting.

The main contributions of the paper are (i) the introduction of sparse partially linear additive models, (ii) the development of an oracle inequality of SPLAM without *any* assumptions on the design matrix that demonstrates SPLAM’s suitability even in high-dimensional settings and indicates its advantage over SpAM when many of the features in model (1) are linear, and (iii) a thorough set of experiments on both synthetic datasets and real problems that show SPLAMs often outperform ℓ_1 regularized GLMs and SpAM in large-scale data sets.

2 The SPLAM Optimization

We approach the challenging combinatorial problem described in (1) through a convex relaxation. For each feature x_j , we perform an M -dimensional basis expansion, $b(x_j) = [b_1(x_j), \dots, b_M(x_j)]$, where $b_1(x_j) = x_j$ and M is typically small. Our main requirement of the basis is that $b_1(x_j)$ models the linear part and that $[b_2(x_j), \dots, b_M(x_j)]$ models the nonlinear part. We use $X \in \mathbb{R}^{N \times pM}$ to denote the design matrix. SPLAM estimates each $f_j(\cdot)$ by a function in the space spanned by $b(\cdot)$, i.e., $f_j(x_j) = b(x_j)\beta_j$, where $\beta_j \in \mathbb{R}^M$.¹ We use β_{j1} to denote the coefficient of the linear basis $b_1(x_j)$, and $\beta_{j,-1}$ for the coefficient vector on the nonlinear bases, $[b_2(x_j), \dots, b_M(x_j)]$. Let $\beta = [\beta_1^T, \dots, \beta_p^T]^T \in \mathbb{R}^{pM}$. Given a convex smooth loss function $L(y, X, \beta)$, SPLAM is formulated as the solution to the following convex program with hierarchical sparsity regularization:

Optimization Problem 1. *SPLAM*

$$\min_{\beta} L(y, X, \beta) + \lambda \Omega^{SPLAM}(\beta) \quad (2)$$

¹For simplicity we ignore the intercept.

where

$$\Omega^{SPLAM}(\beta) = \sum_{j=1}^p [\alpha \|\beta_j\|_2 + (1 - \alpha) \|\beta_{j,-1}\|_2],$$

$\lambda \geq 0$, and $\alpha \in [0, 1]$.

For regression problems, $L(y, X, \beta) = \frac{1}{2N} \|y - X\beta\|_2^2$ is the quadratic loss and for binary classification problems, $L(y, X, \beta) = \frac{1}{N} \sum_i \log(1 + \exp[-y_i \sum_{k=1}^{pM} X_{ik} \beta_k])$ is the logistic loss where $y_i \in \{-1, 1\}$.

The penalty function Ω^{SPLAM} is convex and is an instance of the hierarchical group lasso [19, 20]. When $\alpha = 1$ and an orthogonal basis is chosen, Problem 1 reduces to SpAM in group lasso form. When $\lambda = \tilde{\lambda}/\alpha$ and α is sufficiently small, SPLAM reduces to the Lasso [21] applied to linear features only. When $\alpha = 0$, only the nonlinear terms are penalized (as λ increases we get more linear features, but none are set to zero). When $\alpha \in (0, 1)$, Problem 1 performs both kinds of model selection needed for a GPLAM. The hierarchical group lasso can be solved efficiently by proximal gradient descent [22] as described in [20]. The idea of this algorithm is to modify the standard gradient steps of L by applying the proximal operator of the nondifferentiable penalty, $\lambda \Omega^{SPLAM}(\cdot)$:

$$\beta^{k+1} = \arg \min_{z \in \mathbb{R}^{pM}} \left\{ \frac{1}{2t^k} \|z - (\beta^k - t^k \nabla L(y, X, \beta^k))\|_2^2 + \lambda \Omega^{SPLAM}(z) \right\}, \quad (3)$$

where t^k is a suitable step size. It is known that setting the step size to the reciprocal of the Lipschitz constant of ∇L guarantees convergence [22]. Given a hierarchical sparsity regularization, such as Ω^{SPLAM} , the proximal operator can be very efficiently solved in the dual by a single pass of block coordinate descent [20].

SPLAM has two tuning parameters, λ and α , and typically the problem is solved over a grid of (λ, α) pairs. Our strategy is to generate a complete regularization path with some α fixed starting at the smallest value of λ for which $\hat{\beta}_j = 0$ for $j = 1, \dots, p$. Starting from this value, we decrease λ exponentially. In Section 3.1, we prove that SPLAM is consistent under general conditions for $\alpha = (1 + \sqrt{6})/(1 + 2\sqrt{6})$ and suitably chosen λ .

3 Statistical Theory

In this section, we seek a deeper understanding of the regimes in which SPLAM works well. In Section 3.1, we prove an upper bound on SPLAM's prediction error in the regression setting. This establishes SPLAM as a reliable method even when $p \gg N$ and gives insight into the factors that influence its prediction performance. In Section 3.2, we consider an asymptotic regime that highlights SPLAM's potential statistical advantage over SpAM.

3.1 Oracle Inequality

Oracle inequalities have been proved for the hierarchical group lasso (see, e.g., [23]) that could be applied to SPLAM. These results follow from the unified framework of [24], which gives both oracle inequalities and recovery guarantees for a wide class of estimators based on decomposable regularizers. However, such results (and others of its kind) make potentially strong (and unverifiable) assumptions on the design matrix (e.g., the restricted isometry property [25], the compatibility condition [26, 27], small coherence [28], etc. See [29]). Since SPLAM's design matrix consists of derived features, such assumptions become even more difficult to interpret. There is, however, a different class of oracle inequalities, known as "slow rates," that make *no assumptions* on the design matrix [30]. In addition, despite their name, these inequalities have been shown in some cases to give faster rates of convergence than the more standard "fast rates" [30]. They are particularly useful in situations where the various assumptions made by the fast rate bounds are known not to apply or would be particularly difficult to interpret.

We derive in this section slow rate bounds for SPLAM, thereby giving us an understanding of its statistical performance under no conditions on X . To the best of our knowledge, these are the first such slow rate bounds derived for the hierarchical group lasso.

Suppose $y = \sum_{j=1}^p X_j \beta_j^0 + \epsilon$, where X_j denotes the $N \times M$ matrix after basis expansion for the j th feature, $\beta_j^0 \in \mathbb{R}^M$ is the true vector of coefficients, and $\epsilon \sim N(0, \sigma^2 I_N)$ is a random vector of noise. We will assume in this section that we have orthogonalized each feature's design matrix, i.e., that $\frac{1}{N} X_j^T X_j = I_M$. We should emphasize that this is a concrete statement about the basis chosen for each feature and within our control—it is different in nature from a statement about the relationship between measured features. We describe the sparsity of the vector $\beta^0 \in \mathbb{R}^{pM}$ in two senses: first, whether a feature is at all relevant, $\mathcal{S}_0 = \{j : \beta_j^0 \neq 0\}$ and, second, whether the feature is nonlinear, $\mathcal{N}_0 = \{j : \beta_{j,-1}^0 \neq 0\}$. We also define the set of linear features, $\mathcal{L}_0 = \mathcal{S}_0 \setminus \mathcal{N}_0$.

Letting $\hat{\beta} \in \arg \min_{\beta} \frac{1}{2N} \|y - \sum_j X_j \beta_j\|_2^2 + \lambda \Omega^{SPLAM}(\beta)$ denote a solution of SPLAM, we now present a slow rate for SPLAM's prediction error.

Theorem 1. *If we take $\lambda \geq 2(1 + 2\sqrt{6})\sigma\sqrt{\log p/N}$ and $\alpha = (1 + \sqrt{6})/(1 + 2\sqrt{6})$, then*

$$\frac{1}{N} \|X\hat{\beta} - X\beta^0\|_2^2 \leq 3\lambda \left[\alpha \sum_{j \in \mathcal{L}_0} |\beta_{j1}^0| + \sum_{j \in \mathcal{N}_0} \|\beta_j^0\|_2 \right] \quad (4)$$

holds with probability at least $1 - 4/p$ as long as $\log p \geq M/8$.

Proof. See Appendix B. □

The above theorem implies that for suitably chosen λ , SPLAM's prediction error converges to 0 in probability as $N \rightarrow \infty$ even if we let p grow like e^{N^γ} with $\gamma < 1$ (assuming the sets \mathcal{L}_0 and \mathcal{N}_0 and the coefficients of features in this set remain fixed). It also shows that our error grows linearly both in the number of linear and nonlinear features in the true model. An interesting implication of the theorem is that $\alpha \approx 0.58$ is a theoretically justifiable choice (although better performance may be achievable by tuning α).

When all features are linear ($\mathcal{N}_0 = \emptyset$), this result reduces to the traditional slow rate bound for the Lasso (up to constants) [31]. Such bounds have been improved for the Lasso by careful incorporation of the design matrix [32], and we speculate that similar improvement could be developed here.

3.2 A Comparison to SpAM When All Features Are Linear

We have seen in the previous section that SPLAM is consistent in prediction error in the presence of both linear and nonlinear features even when $p \gg N$. Since SpAM is a special case of SPLAM (with $\alpha = 1$), similar bounds follow easily. A natural question then is whether there is any *statistical* reason to prefer SPLAM over SpAM (aside from the easier interpretation of a GPLAM over a GAM when many features are linear). Intuitively, it seems that when many features are truly linear, SpAM incurs variance for estimating nonlinear terms without a useful reduction in bias; on the other hand, for SPLAM this would not happen assuming a sufficiently large parameter for the nonlinear-specific penalty. We make this intuition more precise by considering a scenario in which SPLAM is consistent whereas SpAM is not, implying that there is indeed a statistical advantage to using SPLAM.

Suppose that all p features are linear with equal coefficients, i.e., $\beta_j^0 = b e_1 \in \mathbb{R}^M$, and consider an asymptotic regime in which p is fixed and $N = pM$ with $M, N \rightarrow \infty$ (note, Theorem 1 does not apply since here $M > 8 \log p$). We assume that all features are orthogonal, i.e. $\frac{1}{N} X^T X = I_N$, so that SpAM has a simple closed-form expression:

$$\hat{\beta}_j^{SpAM} = \left(1 - \frac{\lambda}{\|\frac{1}{N} X_j^T y\|_2} \right)_+ \frac{1}{N} X_j^T y.$$

A several line argument in the supplementary materials establishes that

$$\lim_{N, M \rightarrow \infty} \frac{1}{N} \|X\hat{\beta}^{SpAM} - X\beta^0\|_2^2 \geq \frac{b^2}{1/b^2 + p/\sigma^2} > 0.$$

Table 1: RMSE for synthetic dataset in Section 4.1. Mean RMSE with one standard deviation is shown.

Model	$\sigma^2 = 1$	$\sigma^2 = 2$	$\sigma^2 = 4$	$\sigma^2 = 8$
Lasso	2.9988 (0.0596)	3.1747 (0.0683)	3.4589 (0.0685)	4.0175 (0.0469)
SPLAM	1.0153 (0.0180)	1.4406 (0.0293)	2.0142 (0.0315)	2.8395 (0.0646)
SpAM	1.0189 (0.0194)	1.4442 (0.0308)	2.0221 (0.0313)	2.8463 (0.0652)

Thus, in the asymptotic regime in which one allows the number of basis vectors to grow linearly with N , one finds that the prediction error is bounded away from zero (regardless of the choice of λ). Interestingly, this lower bound matches (up to constants) the upper bound for the group lasso in Theorem 8.1 of [26].

By contrast, consider SPLAM with $\lambda\alpha = 0$ and $\lambda(1 - \alpha) = \infty$ (e.g., take $\alpha \rightarrow 0$ and $\lambda = \alpha^{-1/2}$). With this choice of parameters, it is apparent that $\hat{\beta}_j^{SPLAM} = X_{j1}^T y \cdot e_1$ is simply the least squares solution on the correct set of variables and

$$\frac{1}{N} \|X\hat{\beta}^{SPLAM} - X\beta^0\|_2^2 \rightarrow 0.$$

While assuming that the number of basis functions, M , is growing linearly in N is of course particularly unfavorable to SpAM (indeed, [8] note that M growing like $N^{1/5}$ is a standard choice), it does serve to verify the intuition regarding the statistical cost of incorrectly assuming nonlinearity. In Section 4.1, we perform an empirical investigation showing the various scenarios in which SPLAM, SpAM, and the Lasso each perform best.

4 Experiments

In this section, we report experimental results for SPLAM. For all our experiments, we use cubic splines with 10 knots for basis expansion: $b(x_j) = [x_j, x_j^2, x_j^3, (x_j - x_{j1}^*)^3_+, \dots, (x_j - x_{jm}^*)^3_+]$, where $(\cdot)_+$ represents the non-negative part and the knot x_j^* is chosen from quantiles in the sample. We choose the best parameters on a held-out validation set and report model performance on a test set.

4.1 Synthetic Problem

To illustrate the use of SPLAM, we generate $N = 10,000$ points from the model $y = 2\sin(2x_1) + x_2^2 + \exp(-x_3) + x_4 - 3x_5 + 2.5x_6 + 10x_7 + 2x_8 - 7x_9 + 5x_{10} + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. In this experiment, we create an additional 90 random features (so $p = 100$) and vary the noise variance $\sigma^2 = 1, 2, 4, 8$. The first 3 nonlinear features are generated uniformly in $[-2.5, 2.5]$, and all other features are uniformly in $[0, 1]$.

We compare SPLAM with the Lasso [21] and SpAM [8]. The Lasso builds a linear model with ℓ_1 regularization and SpAM constructs a sparse additive model. For SPLAM, we consider $\alpha \in \{0.05, 0.1, \dots, 0.95, 1\}$. For each method, we consider the full regularization path with 100 λ s spaced evenly on a log scale. In our experiments, this range is sufficient to find the optimal model structure. Best model parameters are chosen using the validation set. Thus, we have the full spectrum of sparse models from a pure linear model to a pure additive model. For all successive experiments, we use the same method to choose parameters.

Accuracy. Table 1 summarizes the experimental results on a 5-fold cross validation. Since we have 3 nonlinear components in the ground truth model, as expected, both SPLAM and SpAM outperform the Lasso. However, as we carefully control the complexity, SPLAM has consistently higher accuracy than SpAM. In fact, SPLAM outperforms SpAM on *every* cross validation set. As the noise level goes up, performance of all models degrades as one would expect. This shows the excellent accuracy of SPLAM in practice when the true model structure is not known a priori. Note that the accuracy improvement of SPLAM compared to SpAM is small because in this case, we have enough data so that it is possible for SpAM to estimate reliably. However, notice that when some of the variables are set to linear, SPLAM can outperform its nonlinear counterpart of SpAM. In addition, we find that both SPLAM and SpAM can always find the correct support on each cross validation set while Lasso sometimes makes mistakes.

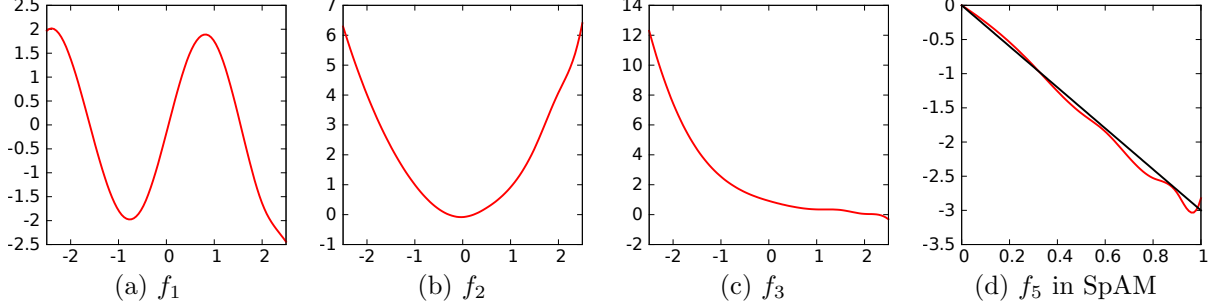


Figure 1: Estimated component functions (in red) for synthetic dataset in Section 4.1. Nonlinear estimates of SPLAM are illustrated in (a) - (c). Figure (d) shows SpAM’s estimate of f_5 , a truly linear component (ground truth shown in black). The wiggleness of SpAM’s estimate is because it does not penalize toward exact linearity as does SPLAM.

Component Estimation. We plot estimated components in Figure 1 for $\sigma^2 = 1$ on a sample of 2,000 points. Figure 1 (a), (b), and (c) visualize the nonlinear components in SPLAM for f_1 , f_2 , and f_3 , respectively. We can see the estimated shape of the component function is very close to the true functions. In this case, SPLAM perfectly recovers which features are linear and nonlinear. For weights on linear components, the relative error is less than 0.1%; we almost get perfect estimation. For comparison, we visualize f_5 in SpAM in red in Figure 1 (d). The ground truth linear function is plotted in black. We can see the component itself is not exactly linear and it overfits to the noise, which explains the degraded predictive performance and the desiderata of recovering linear components exactly. Notice that when setting a component to be exactly linear, we reduce the internal representation complexity; this saves memory and permits fast computation for future predictions.

4.2 Simulation: The Effect of $|\mathcal{N}|$ and $|\mathcal{L}|$ on SPLAM, SpAM, and the Lasso

In this section, we perform a large-scale simulation to gain deeper insights into the Lasso, SPLAM and SpAM. We consider the models with $p = 100$ features: $y = \sum_{j \in \mathcal{L}} x_j + \sum_{j \in \mathcal{N}} \sin(x_j) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$, $\mathcal{L} \cap \mathcal{N} = \emptyset$. We use two parameters γ and δ to control the cardinality of \mathcal{L} and \mathcal{N} , respectively, i.e., $|\mathcal{L}| = \gamma p$ and $|\mathcal{N}| = \delta p$. We choose $\gamma = 0.0, 0.1, \dots, 1.0$ and $\delta = 0.0, 0.1, \dots, 1.0$ ($\gamma + \delta \leq 1$) and for each (γ, δ) pair, we generate 10 different models. For each of those models, we generate N_{train} points for training, N_{valid} points for validation and N_{test} points for testing. We consider three different settings of simulations, $(N_{train}, N_{valid}, N_{test}) = (200, 100, 100), (500, 100, 100), (1000, 200, 200)$. Best model parameters are chosen using the validation set, and model accuracy is evaluated as the average RMSE of 10 models on test sets.

Figure 2 shows the results for the simulations. For each (γ, δ) pair, we plot which model wins on average. It is clear that for pure linear ($\delta = 0$) and pure additive ($\gamma = 0$), SPLAM has no advantage over the Lasso or SpAM.

When $N_{train} = 200$, both SpAM and SPLAM overfit significantly when there are a lot of nonlinear components, since a large number of nonlinear components leads to a large parameter space and this small amount of data is not enough for reliable estimates. The Lasso wins over the other methods on most of the cases by trading off variance for bias. SPLAM outperforms the Lasso in regimes with a mixture of small nonlinear components and a reasonable number of linear components. When we increase the number of data points in the training set ($N_{train} = 500$), more reliable estimates can be obtained so SpAM wins back from the Lasso on cases where we only have nonlinear components (the Lasso having only linear features is incapable of estimating the nonlinear effects present in the data). Interestingly, the Lasso is still the best when there are a lot of nonlinear components since in this regime the data cannot support the large number of parameters for reliable estimation. SPLAM, however, becomes the best on most settings since it can better model the mixture of linear and nonlinear effects when there is enough data. Not surprisingly, when there are

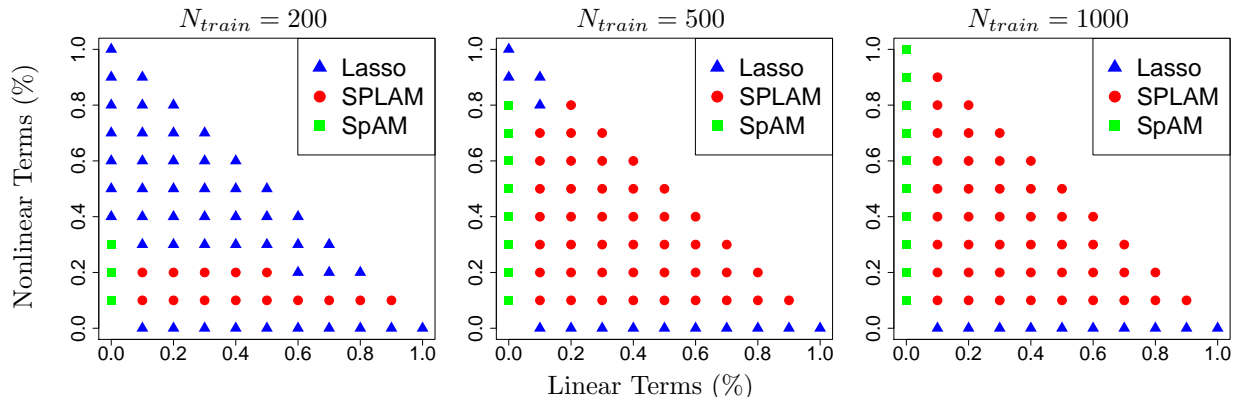


Figure 2: Results of simulation in Section 4.2. Each point shows the winning model for a given (γ, δ) pair.

Table 2: Dataset characteristics and classification error (%). Each cell contains the mean error with one standard deviation.

Dataset	Size	Test	Dimension	%Pos	ℓ_1 -LR	SPLAM	SpAM
Spambase	4601	920	58	39.40	7.38 (0.87)	6.35 (0.82)	6.59 (0.96)
Gisette	6000	1200	5001	50.00	2.52 (0.71)	2.32 (0.62)	2.85 (0.48)
RCV1	697641	418584	47236	52.46	2.66 (0.02)	2.55 (0.01)	2.70 (0.02)
Pantheon	62849	37709	10000	50.00	9.07 (0.10)	9.00 (0.11)	9.25 (0.12)

enough data ($N_{train} = 1000$), SPLAM dominates all cases in which both linear and nonlinear components are present. This is because the Lasso is unable to model nonlinear effects and because SpAM has higher variance than SPLAM without being less biased. We can see that our theoretical results in Section 3 are well-supported by the empirical study in this section.

4.3 Real Problems

In this section, we report experimental results on several real classification problems. We choose datasets with different dimensions and sizes. Table 2 summarizes the characteristics of the datasets and presents the predictive performance of ℓ_1 regularized logistic regression, SPLAM, and SpAM with means and standard deviations on 5-fold cross validations.

Email Classification. We first consider a classification problem for detecting spam emails (Spam-base) [33]. The features measure the percentage of specific words or characters in the email, the average and maximum lengths of uninterrupted upper case letters, and the total number of such letters. We see from Table 2 that by allowing features to act nonlinearly, the error of SpAM decreases substantially compared to ℓ_1 regularized logistic regression. However, by explicitly setting some of the variables to stay linear, SPLAM further outperforms SpAM.

Handwritten Digit Recognition. We use the “Gisette” dataset constructed from NIPS 2003 feature selection challenge [34]. The problem is to separate the highly confusable digits “4” and “9”. For the purpose of the feature selection challenge, pixels were samples at random in the middle top part of the feature containing the information necessary to disambiguate “4” from “9” and higher order features were created as products of these pixels to force the problem in a higher dimensional feature space. Distractor features with no predictive power were also added. Since the dimension of this dataset is significantly larger than the previous dataset while the size of the dataset remains similar, we expect SpAM to overfit as shown in Table 2. In our experiments, the best SpAM model that we can get is always worse than ℓ_1 regularized logistic regression on each cross validation set while our SPLAM outperforms ℓ_1 regularized logistic regression on

most cross validation sets. Our SPLAM selects about 400 features, with about 10 of them being nonlinear and the rest being linear. This confirms that by allowing a small number of features to act nonlinearly, we can further improve the classification performance, and yet by setting most of features as linear, we effectively control the complexity and avoid overfitting.

Text Categorization. Text categorization is an important task for many natural language processing applications. We use Reuters Corpus Volume I (RCV1) which involves binary classification [35]. Table 2 shows the predictive performance of the three models. We see that SPLAM outperforms the others. This suggests that in high dimensions, although ℓ_1 regularized linear model is popular, there is extra accuracy that can be obtained if some features are allowed to be nonlinearly transformed. However, if all features are allowed to be nonlinearly transformed, such as in SpAM, the model will overfit and a suboptimal model is obtained.

Image Matching. Many new computer vision applications are utilizing large-scale datasets of places derived from the many billions of photos on the Web. Image matching is a central procedure to those applications which tests whether two images are geometrically consistent [36]. Since image matching is an expensive procedure, image pairs are usually pre-filtered with a lightweight classification procedure to estimate whether two images are likely to pass the geometric verification. In this study, we use the “Pantheon” dataset in [36]. Each image is represented using bag-of-visual-words model with a vocabulary of 10,000 visual words. From Table 2 we again observe that by carefully controlling the complexity of the model, SPLAM has better predictive performance than the other two models. On average ℓ_1 regularized logistic regression selects 1538 features while SPLAM selects 1529 features with 10 of them being nonlinear.

5 Conclusion

In this paper, we introduce the sparse partially linear additive model that performs two model-selection tasks within a single convex hierarchical sparse regularization problem. This formulation permits an efficient optimization algorithm, making the GPLAM framework practical in machine learning settings. We develop an oracle inequality of SPLAM that makes no assumptions on the design matrix, and we study SPLAM’s advantage over SpAM when many of the features in the model are linear. Our thorough experiments demonstrate that SPLAM can effectively and accurately find relevant components with proper complexity and is very competitive for additive modeling. In particular, on large-scale, high-dimensional datasets, SPLAM improves accuracy from the popular linear model by allowing a small set of features to have a nonlinear effect on the response.

6 Acknowledgments

We thank Johannes Lederer for useful discussions. This research has been supported by the NSF under Grants IIS-0911036 and IIS-1012593. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

A Efficient SPLAM Optimization

Although the standard approach is to use proximal operator, in this supplementary, we consider different (more efficient) approaches using block coordinate gradient descent (BCGD) and block coordinate descent (BCD) to fit SPLAM. We will first describe the general method of solving SPLAM using BCGD in Section A.1. For regression problems, we exploit the property of the quadratic loss and propose a more efficient BCD method in Section A.2. We describe our optimization approach in this section for completeness.

A.1 Block Coordinate Gradient Descent

We first propose a BCGD method to solve SPLAM. For each block j , we form the proximal operator and use its solution as our new estimate of β_j for block j . A natural extension to the proximal gradient method in this BCGD framework is to allow different step sizes for each block j . Thus, the subproblem becomes

$$\beta_j^{k+1} = P_{t_j}^j(\beta^k) \quad (5)$$

where t_j is the step size for block j and

$$P_t^j(\beta) = \arg \min_{z \in \mathbb{R}^M} \left\{ \frac{1}{2t} \|z - (\beta_j - t \nabla_j L(\beta))\|_2^2 + \lambda \alpha \|\beta_j\|_2 + \lambda(1 - \alpha) \|\beta_{j,-1}\|_2 \right\}, \quad (6)$$

and $\nabla_j f(\cdot) \in \mathbb{R}^M$ is the gradient vector on block j . Note that with the BCGD framework, computing the Lipschitz constant C_j for $\nabla_j L$ is no longer expensive; $X_j^T X_j$ is just an M -by- M matrix, where M is typically very small. Thus, we also compute the minimum step size $1/C_j$ to avoid the step size t_j going below this value.

This proximal problem can be solved very efficiently using a primal-dual approach [20]. Let $g_j = \beta_j^k - t_j \nabla_j L(\beta^k)$ and consider the dual problem,

$$\min_{\gamma_1, \gamma_2} \frac{1}{2} \|g_j - \gamma_1 - [0, \gamma_2^T]^T\|_2^2 \quad (7)$$

$$s.t. \quad \|\gamma_1\|_2 \leq t_j \lambda \alpha \quad (8)$$

$$\|\gamma_2\|_2 \leq t_j \lambda (1 - \alpha) \quad (9)$$

where $\gamma_1 \in \mathbb{R}^M$ and $\gamma_2 \in \mathbb{R}^{M-1}$.

As shown in [20], this dual problem can be solved in *one pass* of block coordinate descent as follows,

$$\gamma_2 = \Pi_{t_j \lambda (1 - \alpha)}(g_j, -1) \quad (10)$$

$$\gamma_1 = \Pi_{t_j \lambda \alpha}(g_j - [0, \gamma_2^T]^T) \quad (11)$$

where $\Pi_r(u)$ projects the vector u onto the ball of radius r . The solution to the primal is $z = g_j - \gamma_1 - [0, \gamma_2^T]^T$.

We perform a backtracking line search to ensure the following inequality holds,

$$L(\tilde{\beta}) \leq L(\hat{\beta}) + \langle \beta'_j - \beta_j^k, \nabla_j L(\hat{\beta}) \rangle + \frac{1}{2t_j} \|\beta'_j - \beta_j^k\|_2^2, \quad (12)$$

where

$$\beta'_j = P_{t_j}^j(\beta^k) \quad (13)$$

$$\hat{\beta} = [\beta_1^{k+1T}, \dots, \beta_{j-1}^{k+1T}, \beta_j^{kT}, \dots, \beta_p^{kT}]^T \quad (14)$$

$$\tilde{\beta} = [\beta_1^{k+1T}, \dots, \beta_{j-1}^{k+1T}, \beta'_j{}^T, \dots, \beta_p^{kT}]^T. \quad (15)$$

Algorithm 1 summarizes our block coordinate gradient descent method. We cycle through all blocks (Line 5), solve the proximal operator for that block (Line 7-10) and check if the step size is proper using a backtracking line search (Line 11-14).

A.2 Block Coordinate Descent

Although Algorithm 1 is applicable to any differentiable loss function L (including quadratic loss), we propose a more efficient BCD approach to solve regression problems by exploiting the property of quadratic loss.

Algorithm 1 SPLAM via BCGD

```
1:  $t_j = t_j^0$ , for  $j = 1, \dots, p$ 
2:  $k \leftarrow 0$ 
3:  $\beta^0 \leftarrow 0$ 
4: while not converge do
5:   for  $j = 1$  to  $p$  do
6:     while true do
7:        $g_j \leftarrow \beta_j^k - t_j \nabla_j L(\beta^k)$ 
8:        $\gamma_2 \leftarrow \Pi_{t_j \lambda(1-\alpha)}(g_{j,-1})$ 
9:        $\gamma_1 \leftarrow \Pi_{t_j \lambda \alpha}(g_j - [0, \gamma_2^T]^T)$ 
10:       $\beta_j^{k+1} \leftarrow g_j - \gamma_1 - [0, \gamma_2^T]^T$ 
11:      if Inequality 12 holds then
12:        break
13:      else
14:         $t_j \leftarrow \min(\eta t_j, 1/C_j)$ 
15:       $k \leftarrow k + 1$ 
```

Consider an orthonormal basis expansion where each block j in the design matrix X , in this case denoted as Q_j , is orthonormal, i.e., $Q_j^T Q_j = I_M$. In block coordinate descent, for quadratic loss each subproblem on block j can be formalized using matrix form as follows,

$$\min_{\beta_j} \frac{1}{2N} \|r_j - Q_j \beta_j\|_2^2 + \lambda \alpha \|\beta_j\|_2 + \lambda(1 - \alpha) \|\beta_{j,-1}\|_2 \quad (16)$$

where $r_j = y - \sum_{k \neq j} Q_k \beta_k$ is the partial residual.

Using the orthonormality of Q_j , we observe that this problem has the same minimizer as.

$$\min_{\beta_j} \frac{1}{2N} \|Q_j^T r_j - \beta_j\|_2^2 + \lambda \alpha \|\beta_j\|_2 + \lambda(1 - \alpha) \|\beta_{j,-1}\|_2 \quad (17)$$

Notice that this is the proximal operator applied on $Q_j^T r_j$ (instead of $\beta_j - t \nabla_j L(\beta)$), which can be similarly solved in one pass in the dual form. Thus, we completely eliminate the need to use the step size and the back tracking line search in BCGD, which makes optimization much more efficient.

We perform a QR decomposition for each block j using Gram-Schmidt process, in order to preserve the linear basis in the first column of each block, i.e., $X_j = Q_j R_j$, for any X_j . Algorithm 2 summarizes our BCD algorithm. With suitable choices of $b(x_j)$ (such as the cubic spline bases), such QR decomposition usually works well in practice and R_j is usually invertible.

Note that we are solving a different optimization problem and this solution is not the optimal solution to the original problem. Nevertheless, our choice of feature representation is fairly arbitrary. From the basis expansion's perspective, all that matters is the space spanned by X_j . Our penalty, however, is sensitive to the basis that we are working with so it is important to preserve the linear basis in the first column of each block.

A.3 Convergence

While it is apparent that Algorithm 2 is the standard block coordinate descent algorithm with guaranteed convergence, it is not obvious that Algorithm 1 also converges since we are running proximal operator at the block level. In this section, we show that Algorithm 1 fits the general BCGD framework [37] and therefore the global convergence is guaranteed. A similar convergence result for group lasso has also been shown [38]. We first briefly review the general BCGD algorithm.

Algorithm 2 SPLAM via BCD

```

1:  $\beta^0 \leftarrow 0$ 
2: while not converge do
3:   for  $j = 1$  to  $p$  do
4:      $r_j \leftarrow y - \sum_{k \neq j} Q_k \beta_k$ 
5:      $g_j \leftarrow Q_j^T r_j$ 
6:      $\gamma_2 \leftarrow \Pi_{t_j \lambda (1-\alpha)}(g_j, -1)$ 
7:      $\gamma_1 \leftarrow \Pi_{t_j \lambda \alpha}(g_j - [0, \gamma_2^T]^T)$ 
8:      $\beta_j \leftarrow g_j - \gamma_1 - [0, \gamma_2^T]^T$ 

```

Let $F(\beta) = L(\beta) + h(\beta)$, where $h(\beta) = \lambda \Omega^{SPLAM}(\beta)$. At each iteration k , for block j , choose a symmetric matrix H^k , and compute the search direction.

$$d^k = \arg \min_d \{ \nabla L(\beta^k)^T d + \frac{1}{2} d^T H^k d + h(\beta^k + d) \} \quad (18)$$

where $\forall i \notin \mathcal{G}_j, d_i = 0$. Then a step size $\alpha^k > 0$ is chosen so that the following Armijo rule is satisfied,

$$F(\beta^k + \alpha^k d^k) \leq F(\beta^k) + \alpha^k \sigma \Delta^k \quad (19)$$

where $0 < \sigma < 1, 0 \leq \gamma < 1$, and

$$\Delta^k \stackrel{\text{def}}{=} \nabla L(\beta^k)^T d^k + \gamma d^{k^T} H^k d^k + h(\beta^k + d^k) - h(\beta^k), \quad (20)$$

Once the step size α^k is determined, update $\beta^{k+1} = \beta^k + \alpha^k d^k$.

Theorem 2 in [37] guarantees the global convergence when $\bar{\theta}I \succeq H^k \succeq \underline{\theta}I, 0 < \underline{\theta} \leq \bar{\theta}$.

Theorem 2. *Algorithm 1 fits the general BCGD framework [37]. The global convergence is guaranteed and Algorithm 1 converges Q-linearly.*

Proof. First, for block j , setting $H^k = \frac{1}{t_j}I$, Equation 18 is equivalent to our proximal operator for block j after ignoring constants. Next, notice that when $\alpha^k = 1, \sigma = 1$, and $\gamma = \frac{1}{2}$, the Armijo rule becomes our backtracking line search step in Inequality 12. That is, the effort of choosing step size is shifted to finding H^k . Besides, Lemma 1 in [37] suggests $\nabla L(\beta^k)^T d^k + d^{k^T} H^k d^k + h(\beta^k + d^k) - h(\beta^k) \leq 0$. Since $H^k \succ 0$, with $\gamma = \frac{1}{2}$, we can easily see $\Delta^k \leq 0$ whenever $d^k \neq 0$, which means if the Armijo rule holds for $\sigma = 1$, it must also hold for $\sigma < 1$. Finally, we show that $\bar{\theta}I \succeq H^k \succeq \underline{\theta}I$. Assume the initial step size is t_j^0 , this is true when $\bar{\theta} = \max\{C_j, 1/t_j^0\}$ and $\underline{\theta} = \min\{C_j, 1/t_j^0\}$. Thus, according to Theorem 2 in [37], Algorithm 1 converges Q-linearly. \square

A.4 Practical Issues

A.4.1 Active Set Strategy

We employ the widely used active set strategy [39, 40, 41]. After a complete cycle through all the variables, we iterate only on the active set till convergence. If another complete cycle does not change the active set, we are done, otherwise the process is repeated.

A.4.2 Regularization Path

Similar to `glmnet` [39], the optimization of SPLAM also uses two parameters, λ and α , which usually involves a grid search on values of (λ, α) pairs. Our strategy is to generate a complete regularization path with some α fixed, and therefore we need to find the smallest value λ_{max} for which $\beta_j = 0$ for $j = 1, \dots, p$. Starting from λ_{max} , we decrease λ exponentially.

Algorithm 3 Finding λ_{max}

```
1:  $\lambda_h \leftarrow \max_j \frac{\|\nabla_j L(0)\|_2}{\alpha}$ 
2:  $\lambda_l \leftarrow 0$ 
3: while  $\lambda_h - \lambda_l \geq \epsilon$  do
4:    $\lambda \leftarrow \frac{\lambda_h + \lambda_l}{2}$ 
5:   if  $\forall j, P_t^j(0) = 0$  then
6:      $\lambda_h \leftarrow \lambda$ 
7:   else
8:      $\lambda_l \leftarrow \lambda$ 
9:  $\lambda_{max} = \lambda_h$ 
```

The key question is how to find λ_{max} . Notice that for all $\lambda \geq \lambda_{init} = \max_j \frac{\|\nabla_j L(0)\|_2}{\alpha}$, zero point will be the solution to our optimization problem. Therefore, we perform a binary search to find λ_{max} . As described in Algorithm 3, we start with λ_{init} (Line 1) and effectively shrink the interval $[\lambda_l, \lambda_h]$ (Line 3 - 8) to locate λ_{max} .

B Proof of Theorem 1 in Main Paper

Proof. By definition of $\hat{\beta}$,

$$\frac{1}{2N} \|y - X\hat{\beta}\|_2^2 + \lambda \Omega^{SPLAM}(\hat{\beta}) \leq \frac{1}{2N} \|y - X\beta^0\|_2^2 + \lambda \Omega^{SPLAM}(\beta^0),$$

which after some algebra (writing $\hat{\Delta} = \hat{\beta} - \beta^0$) leads to

$$\frac{1}{2N} \|X\hat{\Delta}\|_2^2 + \lambda \Omega^{SPLAM}(\hat{\beta}) \leq \frac{1}{N} \epsilon^T X\hat{\Delta} + \lambda \Omega^{SPLAM}(\beta^0) \quad (21)$$

Define the empirical process as,

$$V_N(\hat{\Delta}) = \frac{1}{N} \epsilon^T X\hat{\Delta} = \frac{1}{\sqrt{N}} \sum_{j=1}^p V_j^T \hat{\Delta}_j \quad (22)$$

where $V_j = \frac{1}{\sqrt{N}} X_j^T \epsilon \in \mathbb{R}^M$.

Now we bound the empirical process. First we notice that,

$$|V_j^T \hat{\Delta}_j| \leq \frac{1}{2} \left[|V_j^T \hat{\Delta}_j| + |V_{j1} \hat{\Delta}_{j1}| + |V_{j,-1}^T \hat{\Delta}_{j,-1}| \right] \quad (23)$$

$$\leq \frac{1}{2} \left[\|V_j\|_2 \|\hat{\Delta}_j\|_2 + |V_{j1}| |\hat{\Delta}_{j1}| + \|V_{j,-1}\|_2 \|\hat{\Delta}_{j,-1}\|_2 \right] \quad (24)$$

Thus $|V_N(\hat{\Delta})|$ can be bounded as follows.

$$|V_N(\hat{\Delta})| \leq \frac{1}{\sqrt{N}} \sum_{j=1}^p |V_j^T \hat{\Delta}_j| \quad (25)$$

$$\leq \frac{1}{2\sqrt{N}} \left(\max_j \|V_j\|_2 \|\hat{\Delta}\|_{2,1} + \max_j |V_{j1}| \sum_j |\hat{\Delta}_{j1}| + \max_j \|V_{j,-1}\|_2 \|\hat{\Delta}_{\cdot,-1}\|_{2,1} \right) \quad (26)$$

$$\leq \frac{1}{2\sqrt{N}} \left[(\max_j \|V_j\|_2 + \max_j |V_{j1}|) \|\hat{\Delta}\|_{2,1} + \max_j \|V_{j,-1}\|_2 \|\hat{\Delta}_{\cdot,-1}\|_{2,1} \right] \quad (27)$$

Observing that $V_j \sim N(0, \sigma^2 I_M)$, we have $\|V_j\|_2^2 \sim \sigma^2 \chi_M^2$. Thus, by Lemma 6.2 and 8.1 of [26], we have

$$P\left(\frac{\max_j |V_{j1}|}{2\sqrt{N}} > \nu_1\right) \leq 2e^{-x} \quad (28)$$

$$P\left(\frac{\max_j \|V_j\|_2}{2\sqrt{N}} > \nu_2\right) \leq e^{-x} \quad (29)$$

$$(30)$$

where,

$$\nu_1^2 = \frac{\sigma^2}{2N}(x + \log p) \quad (31)$$

$$\nu_2^2 = \frac{\sigma^2}{4N} \left[M + \sqrt{4M(x + \log p)} + 4(x + \log p) \right] \quad (32)$$

Thus, we have

$$P\left(\frac{\max_j \|V_j\|_2 + \max_j |V_{j1}|}{2\sqrt{N}} > \nu_1 + \nu_2\right) \leq 3e^{-x} \quad (33)$$

$$P\left(\frac{\max_j \|V_{j,-1}\|_2}{2\sqrt{N}} > \nu_2\right) \leq e^{-x} \quad (34)$$

Therefore (with union bound),

$$P\left(|V_N(\hat{\Delta})| \leq \left[(\nu_1 + \nu_2)\|\hat{\Delta}\|_{2,1} + \nu_2\|\hat{\Delta}_{\cdot,-1}\|_{2,1}\right]\right) \geq 1 - (e^{-x} + 3e^{-x}) \quad (35)$$

$$= 1 - 4e^{-x} \quad (36)$$

Thus, by (21) we have with probability at least $1 - 4e^{-x}$ that

$$\frac{1}{2N}\|X\hat{\Delta}\|_2^2 + \lambda\Omega^{SPLAM}(\hat{\beta}) \leq (\nu_1 + \nu_2)\|\hat{\Delta}\|_{2,1} + \nu_2\|\hat{\Delta}_{\cdot,-1}\|_{2,1} + \lambda\Omega^{SPLAM}(\beta^0) \quad (37)$$

Let $\lambda_1 = \lambda\alpha$ and $\lambda_2 = \lambda(1 - \alpha)$, we can take $\lambda_1 = 2(\nu_1 + \nu_2)$ and $\lambda_2 = 2\nu_2$. Thus, (37) implies

$$\frac{1}{2N}\|X\hat{\Delta}\|_2^2 \leq (\lambda/2)\Omega^{SPLAM}(\hat{\Delta}) - \lambda\Omega^{SPLAM}(\hat{\beta}) + \lambda\Omega^{SPLAM}(\beta^0) \quad (38)$$

$$\leq (\lambda/2) \left[\Omega^{SPLAM}(\hat{\beta}) + \Omega^{SPLAM}(\beta^0) \right] - \lambda\Omega^{SPLAM}(\hat{\beta}) + \lambda\Omega^{SPLAM}(\beta^0) \quad (39)$$

$$= (3\lambda/2)\Omega^{SPLAM}(\beta^0) - (\lambda/2)\Omega^{SPLAM}(\hat{\beta}) \quad (40)$$

by the triangle inequality. Thus,

$$\frac{1}{N}\|X\hat{\Delta}\|_2^2 \leq 3\lambda\Omega^{SPLAM}(\beta^0). \quad (41)$$

By choosing $x = \log p$, we can ensure our inequality holds with probability at least $1 - 4/p$. This means,

$$\nu_1^2 = \frac{\sigma^2}{N} \log p \quad (42)$$

$$\nu_2^2 = \frac{\sigma^2}{4N} \left[M + \sqrt{8M \log p} + 8 \log p \right]. \quad (43)$$

Define $\tilde{\nu}_1^2 \stackrel{\text{def}}{=} \frac{\sigma^2}{N} \log p$ and notice that $\nu_2^2 \leq 6\sigma^2 \log p / N \stackrel{\text{def}}{=} \tilde{\nu}_2^2$ if $\log p \geq M/8$. Now, as long as $\log p \geq M/8$, we can take $\lambda \geq 2(\tilde{\nu}_1 + 2\tilde{\nu}_2) = 2(1 + 2\sqrt{6})\sigma\sqrt{\log p / N}$ and

$$\alpha = \frac{\tilde{\nu}_1 + \tilde{\nu}_2}{\tilde{\nu}_1 + 2\tilde{\nu}_2} = \frac{1 + \sqrt{6}}{1 + 2\sqrt{6}}, \quad (44)$$

with probability at least $1 - 4/p$, we have

$$\frac{1}{N} \|X \hat{\Delta}\|_2^2 \leq 3\lambda \Omega^{SPLAM}(\beta^0). \quad (45)$$

Finally, observe that

$$\Omega^{SPLAM}(\beta^0) = \alpha \sum_{j \in S^0} \|\beta_j^0\|_2 + (1 - \alpha) \sum_{j \in N^0} \|\beta_{j,-1}^0\|_2 \quad (46)$$

$$\leq \alpha \sum_{j \in L^0} \|\beta_j^0\|_2 + \sum_{j \in N^0} \|\beta_j^0\|_2 \quad (47)$$

$$= \alpha \sum_{j \in L^0} |\beta_{j1}^0| + \sum_{j \in N^0} \|\beta_j^0\|_2 \quad (48)$$

□

C Proof of Lower Bound on SpAM's Prediction Error

We assume that all p features are linear with equal coefficients, i.e., $\beta_j^0 = be_1 \in \mathbb{R}^M$ and consider an asymptotic regime in which p is fixed and $N = pM$, with $M, N \rightarrow \infty$. We assume that all features are orthogonal, i.e., $\frac{1}{N} X^T X = I_{pM}$. In the main paper, we note that SpAM in this case is given by the expression:

$$\hat{\beta}_j^{SpAM} = \gamma_j(\lambda) \frac{1}{N} X_j^T y \quad \text{where} \quad \gamma_j(\lambda) = \left(1 - \frac{\lambda}{\|\frac{1}{N} X_j^T y\|_2} \right)_+.$$

Now, $\frac{1}{N} X_j^T y = be_1 + U_j$ where $U_j = \frac{1}{N} X_j^T \epsilon \sim N(0, \frac{\sigma^2}{N} I_M)$. Since $\|\frac{1}{N} X_j^T y\|_2^2 \rightarrow b^2 + \sigma^2/p$, asymptotically, the shrinkage factor $\gamma_j(\lambda) = \gamma$ is a nonrandom value, not depending on j , and the prediction error is

$$\begin{aligned} \frac{1}{N} \|X \hat{\beta}^{SpAM} - X \beta^0\|^2 &= \sum_{j=1}^p \|\gamma(be_1 + U_j) - be_1\|^2 \\ &= \gamma^2(b^2 p + \sum_{j=1}^p [\|U_j\|^2 + 2bU_{j1}]) + pb^2 - 2\gamma \sum_{j=1}^p b(b + U_{j1}) \\ &\rightarrow \gamma^2(b^2 p + \sigma^2) + pb^2 - 2\gamma pb^2. \end{aligned}$$

For the best possible asymptotic error, we can choose $\gamma = pb^2/(pb^2 + \sigma^2)$ (equivalent to choosing the best λ). At this value,

$$\lim_{N \rightarrow \infty} \frac{1}{N} \|X \hat{\beta}^{SpAM} - X \beta^0\|^2 \geq \frac{b^2}{1/b^2 + p/\sigma^2} > 0.$$

Thus, SpAM is not consistent in terms of prediction error in this asymptotic regime.

To see that SPLAM with $\lambda\alpha = 0$ and $\lambda(1 - \alpha) = \infty$ is consistent in terms of prediction error, observe that $\hat{\beta}^{SPLAM} = (X_{j1}^T y)e_1 = (b + U_{j1})e_1$ and

$$\frac{1}{N} \|X \hat{\beta}^{SPLAM} - X \beta^0\|^2 = \sum_{j=1}^p \|(b + U_{j1})e_1 - be_1\|^2 = \sum_{j=1}^p U_{j1}^2 \sim \frac{\sigma^2}{N} \chi_p^2 \rightarrow 0.$$

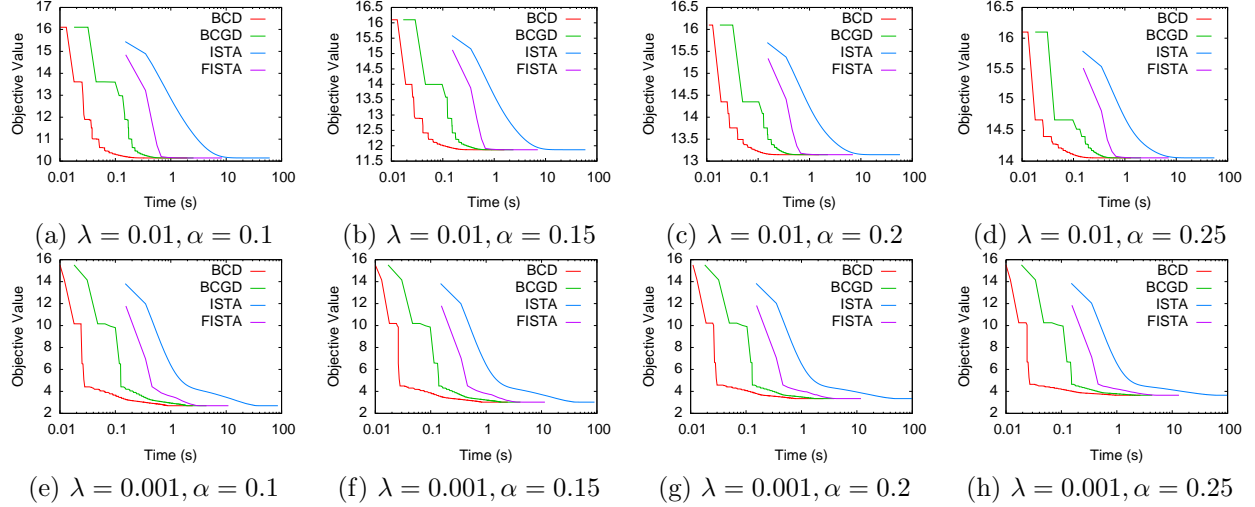


Figure 3: Objective value vs. running time for synthetic dataset in Section 4.1.

D Experiments

In this section, we compare our BCGD algorithm and BCD algorithm with ISTA and FISTA [22] using the synthetic function in Section 4.1. We report running time of all the methods on a single core. For BCGD, ISTA, and FISTA, we start with a same initial step size. For fair comparison, we turn off the active set strategy in BCGD and BCD, and we directly use the design matrix after QR decomposition so that all methods are applied to the same optimization problem.

Figure 3 illustrates the running time for all methods using the same synthetic dataset in Section 4.1 for different combinations of λ and α . As expected, FISTA converges much faster than ISTA. However, the BCGD algorithm is faster than both of the these methods. This is because BCGD uses more information than the first-order methods. In addition, we can see that BCD further speeds up the optimization since there is no step size in BCD; this not only solves exactly the subproblem but also avoids the possibility of dampening the step size and repeating the computation on the same block.

References

- [1] W. Härdle and H. Liang. *Partially linear models*. Springer, 2007.
- [2] J. A. Nelder and R. W. M. Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):370–384, 1972.
- [3] T. Hastie and R. Tibshirani. *Generalized additive models*. Chapman & Hall/CRC, 1990.
- [4] R.F. Engle, C.W.J. Granger, J. Rice, and A. Weiss. Semiparametric estimates of the relation between weather and electricity sales. *Journal of the American Statistical Association*, 81(394):310–320, 1986.
- [5] P.J. Green and B.W. Silverman. *Nonparametric regression and generalized linear models: a roughness penalty approach*. CRC Press, 1993.
- [6] H. Lian, X. Chen, and J. Yang. Identification of partially linear structure in additive models with an application to gene expression prediction from sequences. *Biometrics*, 68(2):437–445, 2012.
- [7] G.E. Dinse and S.W. Lagakos. Regression analysis of tumour prevalence data. *Applied Statistics*, 32(3):236–248, 1983.

- [8] P. Ravikumar, H. Liu, J. Lafferty, and L. Wasserman. Sparse additive models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(5):1009–1030, 2009.
- [9] R. Chen, H. Liang, and J. Wang. Determination of linear components in additive models. *Journal of Nonparametric Statistics*, 23(2):367–383, 2011.
- [10] J. Huang, F. Wei, and S. Ma. Semiparametric regression pursuit. *Statistica Sinica*, 22(4):1403, 2012.
- [11] P. Du, G. Cheng, and H. Liang. Semiparametric regression models with additive nonparametric components and high dimensional parametric components. *Computational Statistics & Data Analysis*, 56(6):2006–2017, 2012.
- [12] Y. Lin and H.H. Zhang. Component selection and smoothing in multivariate nonparametric regression. *The Annals of Statistics*, 34(5):2272–2297, 2006.
- [13] H.H. Zhang, G. Cheng, and Y. Liu. Linear or nonlinear? automatic structure discovery for partially linear models. *Journal of the American Statistical Association*, 106(495), 2011.
- [14] F. Bunea. Consistent covariate selection and post model selection inference in semiparametric regression. *The Annals of Statistics*, 32(3):898–927, 2004.
- [15] H. Xie and J. Huang. Scad-penalized regression in high-dimensional partially linear models. *The Annals of Statistics*, pages 673–696, 2009.
- [16] P. Müller and S. van de Geer. The partial linear model in high dimensions. *arXiv preprint arXiv:1307.1067*, 2013.
- [17] H. Lian and H. Liang. Generalized additive partial linear models with high-dimensional covariates. *Econometric Theory*, 29:1136–1161, 12 2013.
- [18] L. Wang, L. Xue, A. Qu, and H. Liang. Estimation and model selection in generalized additive partial linear models for correlated data with diverging number of covariates. *The Annals of Statistics*, 42(2):592–624, 2014.
- [19] P. Zhao, G. Rocha, and B. Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *The Annals of Statistics*, 37(6A):3468–3497, 2009.
- [20] R. Jenatton, J. Mairal, F.R. Bach, and G.R. Obozinski. Proximal methods for sparse hierarchical dictionary learning. In *ICML*, 2010.
- [21] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.
- [22] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.
- [23] S. Chatterjee, K. Steinhaeuser, A. Banerjee, S. Chatterjee, and A. Ganguly. Sparse group lasso: Consistency and climate applications. In *SDM*, 2012.
- [24] S.N. Negahban, P. Ravikumar, M.J. Wainwright, and B. Yu. A unified framework for high-dimensional analysis of m -estimators with decomposable regularizers. *Statistical Science*, 27(4):538–557, 2012.
- [25] E. Candes and T. Tao. The dantzig selector: Statistical estimation when p is much larger than n . *The Annals of Statistics*, pages 2313–2351, 2007.
- [26] P. Bühlmann and S. van de Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer, 2011.

- [27] S. van de Geer. The deterministic lasso. In *JSM*, 2007.
- [28] E.J. Candès and Y. Plan. Near-ideal model selection by ℓ_1 minimization. *The Annals of Statistics*, 37(5A):2145–2177, 2009.
- [29] S. van de Geer and P. Bühlmann. On the conditions used to prove oracle results for the lasso. *Electronic Journal of Statistics*, 3:1360–1392, 2009.
- [30] A.S. Dalalyan, M. Hebiri, and J. Lederer. On the prediction performance of the lasso. *arXiv preprint arXiv:1402.1700*, 2014.
- [31] P. Rigollet and A. Tsybakov. Exponential screening and optimal rates of sparse estimation. *The Annals of Statistics*, 39(2):731–771, 2011.
- [32] M. Hebiri and J. Lederer. How correlations influence lasso prediction. *IEEE Transactions on Information Theory*, 59(3):1846–1854, 2013.
- [33] T. Hastie, R. Tibshirani, and J.H. Friedman. *The elements of statistical learning (2nd edition)*. Springer New York, 2009.
- [34] <http://www.nipsfsc.ecs.soton.ac.uk/>.
- [35] D.D. Lewis, Y. Yang, T.G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 5:361–397, 2004.
- [36] Y. Lou, N. Snaveley, and J. Gehrke. Matchminer: Efficient spanning structure mining in large image collections. In *ECCV*, 2012.
- [37] P. Tseng and S. Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117(1-2):387–423, 2009.
- [38] Z. Qin, K. Scheinberg, and D. Goldfarb. Efficient block-coordinate descent algorithms for the group lasso. *Mathematical Programming Computation*, pages 1–27, 2010.
- [39] J.H. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1, 2010.
- [40] B. Krishnapuram, L. Carin, M.A. Figueiredo, and A.J. Hartemink. Sparse multinomial logistic regression: Fast algorithms and generalization bounds. *Transactions on Pattern Analysis and Machine Intelligence*, 27(6):957–968, 2005.
- [41] L. Meier, S. van de Geer, and P. Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71, 2008.